p(B|A)yesgroup.ru

SAMSUNG AI Center - Moscow

Variance Networks

When Expectation Does Not Meet Your Expectations

Kirill Neklyudov*, Dmitry Molchanov*, Arsenii Ashukha*, Dmitry Vetrov

Paper bit.ly/2Jalp7Z

Code bit.ly/2DJDwOD

Motivation

$$w \sim \mathcal{N}(\mu, \sigma^2)$$

Stochastic Networks

$w \sim \mathcal{N}(\mu, 0)$ — Usual Networks

It works! $w \sim \mathcal{N}(0, \sigma^2)$ — This paper: Variance Nets

We can store information using variances only!

Stochastic Deep Neural Networks

How to train:

1. $\hat{W} \sim q(W \mid \phi)$ e.g., Gaussian $q(W \mid \phi) = \mathcal{N}(W \mid \phi)$
2. $\nabla_\phi L \cong \nabla_\phi(-\log p(Y \mid X, \hat{W}) + R(\phi))$
3. Update ϕ and repeat until convergence

How to predict:

1. Weight Scaling Rule (WSR) **(heuristic)**

$$p(y \mid x) \approx p(y|x, \mathbb{E}W)$$

\+ Fast and usually works well in practice
\- May yield arbitrarily bad predictions!

2. Monte-Carlo estimate **(proper way)**

$$p(y \mid x) \approx \frac{1}{K}\sum_k p(y|x, \hat{W}_k), \qquad \hat{W}_k \sim q(W|\phi)$$

\+ Produces a correct unbiased estimate
\- Requires to compute the output K times

Variance Networks

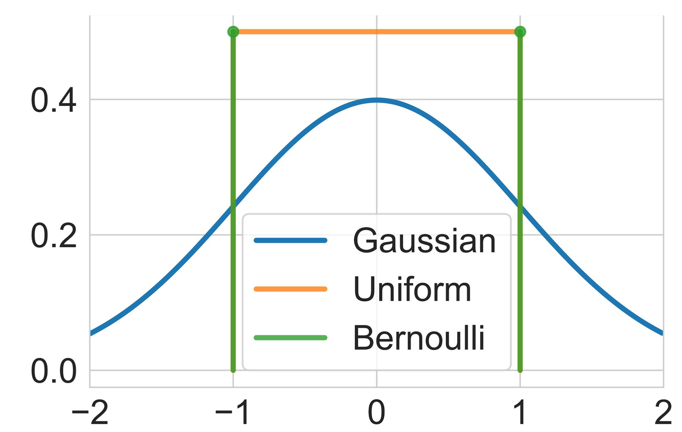
Variance layer has a symmetric weight distribution:

$$q(W \mid \phi) = q(-W \mid \phi)$$

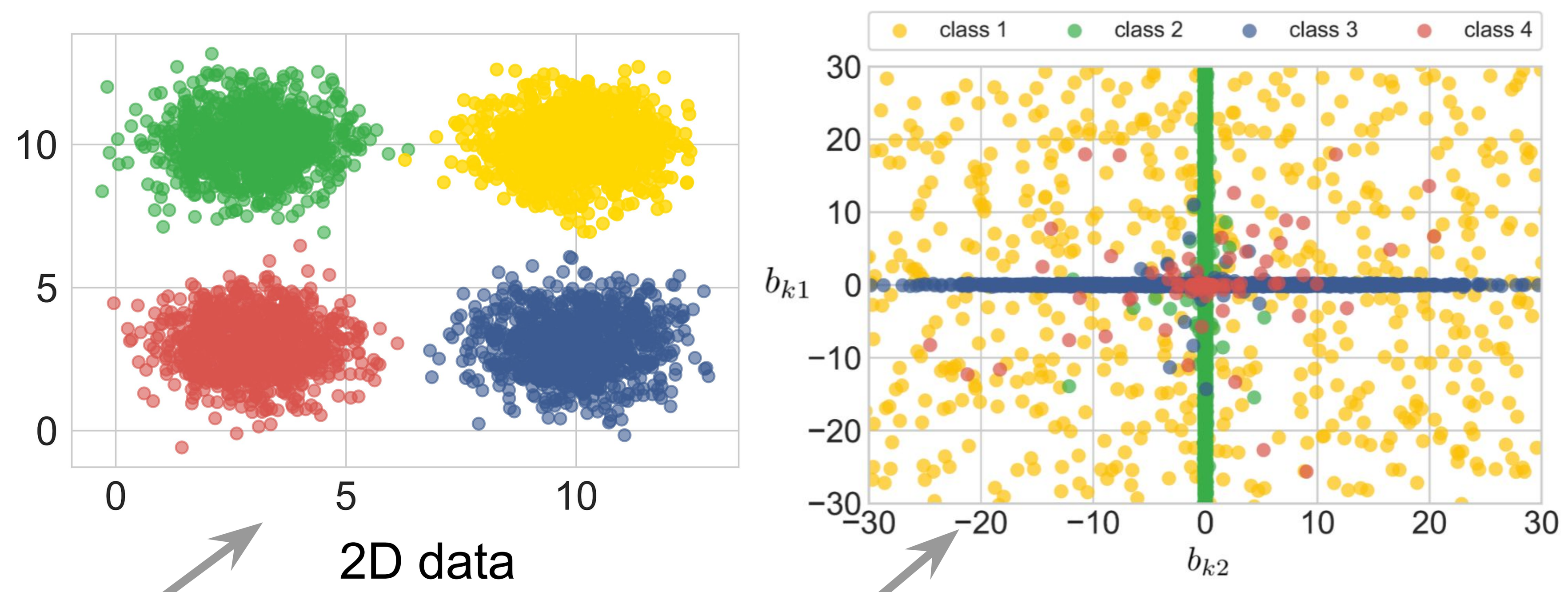
Examples of variance layers:

1. Gaussian: $w_{ij} \sim \sigma_{ij} \cdot \varepsilon_{ij}, \quad \varepsilon_{ij} \sim \mathcal{N}(0,1)$
2. Bernoulli: $w_{ij} \sim \phi_{ij} \cdot (2\varepsilon_{ij} - 1), \quad \varepsilon_{ij} \sim Bernoulli(\frac{1}{2})$
3. Uniform: $w_{ij} \sim \phi_{ij} \cdot \varepsilon_{ij}, \quad \varepsilon_{ij} \sim \mathcal{U}(-1,1)$

$\mathbb{E}W \equiv 0$, hence WSR fails for such distributions!



How it works? A toy example



X → NN → Variance Layer with 2 neurons → NN → Y

Classification results

Architecture	Dataset	Network	Accuracy (%) 1 samp.	Det.	20 samp.
LeNet5	MNIST	Dropout	99.1	99.4	99.4
		Variance	98.2	11.3	99.3
VGG-like	CIFAR10	Dropout	91.0	93.1	93.4
		Variance	91.3	10.0	93.4
VGG-like	CIFAR100	Dropout	77.5	79.8	81.7
		Variance	76.9	5.0	82.2

We achieve the same performance as usual networks!

Variational Dropout→Variance Networks

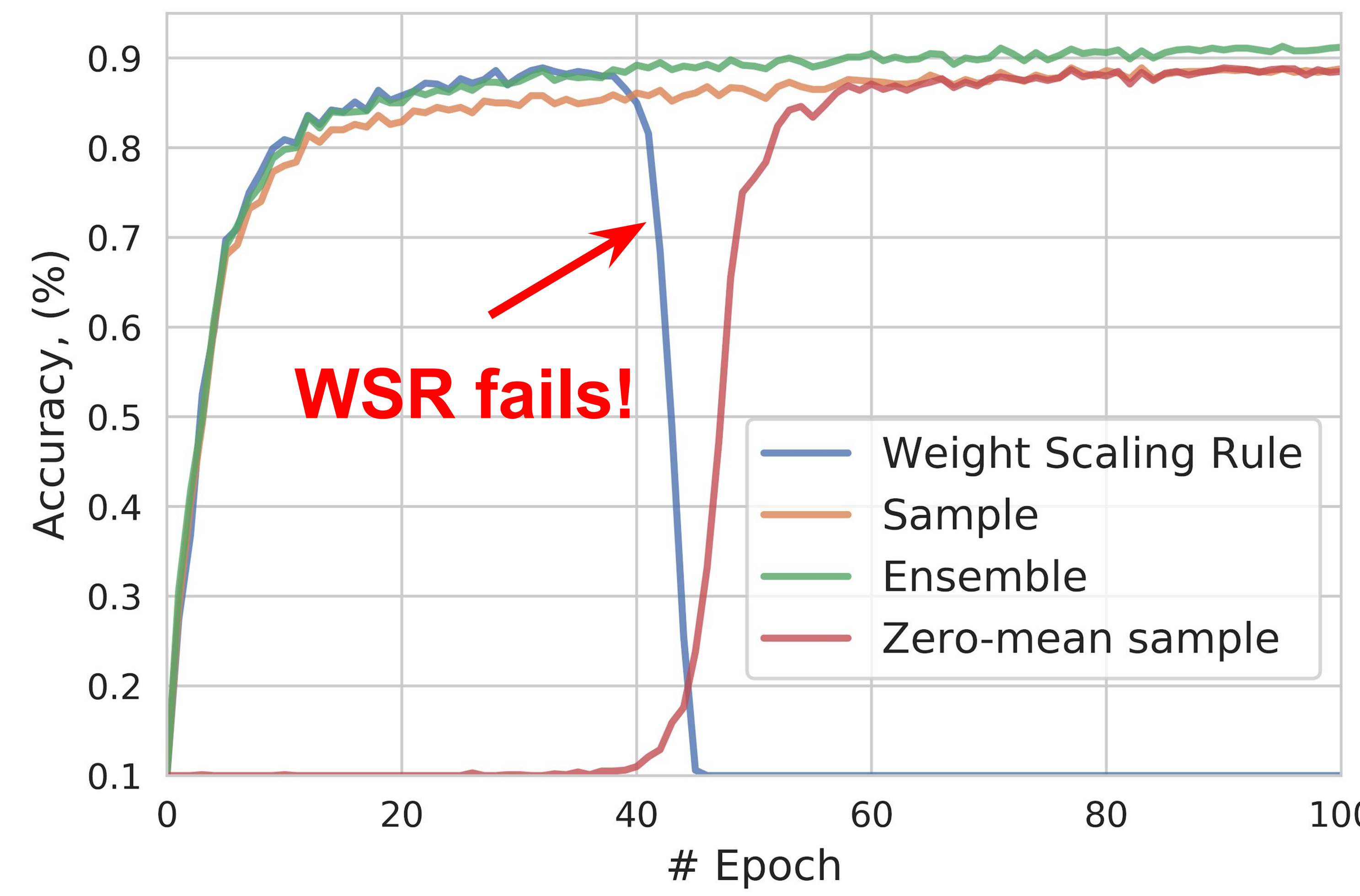
Variational Dropout:

$$\underbrace{-\mathbb{E}_{q(W \mid \phi)} \log p(Y \mid X, W)}_{\text{Data-term (e.g. cross-entropy loss)}} + \underbrace{\mathrm{D_{KL}}(q(W \mid \phi) \,\|\, p(W))}_{\text{Regularizer}} \to \min_\phi$$

$$\tilde{w}_{ij} \sim \mathcal{N}(w_{ij}, \alpha_{ij} w_{ij}^2) \quad p(\tilde{w}_{ij}) \propto \frac{1}{|\tilde{w}_{ij}|} \quad \phi_{ij} = \{\alpha_{ij}, \mu_{ij}\}$$

In practice the Variational Dropout model converges to variance networks!

$$\mathcal{N}(\mu_{ij}, \alpha\mu_{ij}^2) \xrightarrow{\alpha \to \infty} \mathcal{N}(0, \alpha\mu_{ij}^2)$$



Better ELBO

layer-wise $\mathcal{N}(\mu_{ij}, \alpha\mu_{ij}^2)$ — neuron-wise $\mathcal{N}(\mu_{ij}, \alpha_j\mu_{ij}^2)$ — weight-wise $\mathcal{N}(\mu_{ij}, \alpha_{ij}\mu_{ij}^2)$ — additive $\mathcal{N}(\mu_{ij}, \sigma_{ij}^2)$

	Layer	Neuron	Weight	Additive
ELBO	**−9.4**	**−11.0**	−287.4	−227.9
Data term	−9.04	−8.34	−21.13	−31.2
KL term	0.36	2.66	266.25	196.74
Mean prop. acc.	11.3	11.3	96.6	99.2
Test-time averaging	99.3	99.2	99.4	99.2

Less flexible posterior approximations result in much better ELBO!